Course Project of Pratical Machine Learning

Chun-Sheng Wu

Introduction

The objective of this study was to classify subjects' behavior using the data generated by wearable meters. The dataset used in this study can be accessed from here (http://groupware.les.inf.puc-rio.br/har). The data in the dataset were generated from a experiment. Subjects weared meters which could record their body movement, and performed 5 types of behavior during the experiment. The behavoir subject had done and the correspoding data generated by wearable meters were collected in the dataset. 5 behavioral types were indicated by "classe" column in the dataset, with value A~E respectively. Using the dataset metioned above, I tried to build a classifier which could recognize subject's behavior during they are performing behavior.

Data Preparation

In this study, the first step of preparing data was to pick out the data which had the same "form" as the testing data. There were two types of data entries in the training dataset. One type had "yes" value in 'new window' column, and the other type had the value "no". The difference betwee "yes" and "no" type data entry was: "No-type" entry had many columns whose values were all NA. However, "yes-type" entry usually had normal values in those columns. Since the data entries in the testing dataset were all "no-type", I first pick up the "no-type" data in the training dataset. Then, I removed all the columns that contained only NA value.

Second, I removed those columns whose value had no logical relation with the "classe" variable we wanted to predict. These columns were all indication columns, which were: raw_timestamp_part_1, raw_timestamp_part_2, cvtd_timestamp, num_window, new_window, user_name , and X(entry's id).

The whole process described above could be done in the following chunks. In result, there were 19216 entries, 52 varaibles left for the modeling process. All variables contained 0 missing value.

```
clean<-function(data){
  colNames<-names(data)
  dropCol<-c('raw_timestamp_part_1','raw_timestamp_part_2','cvtd_timestamp','num_w
indow','new_window','user_name','classe','X')
  preCol<-colNames[!colNames %in% dropCol]
  data2<-as.data.frame(apply(data[,preCol],2,as.numeric))
  dropCol2<-names(data2)[sapply(data2, function(x) sum(is.na(x))==dim(data2)[1])]
  preCol2<-names(data2)[!names(data2) %in% dropCol2]
  data2<-data2[,preCol2]
  return(data2)
}

data<-read.csv('data/training.csv',na.strings =c('NA',"",'#DIV/0!'))
data<-data[data$new_window=='no',]
classe<-data$classe
data2<-clean(data)
```

Modeling

I applied conditional inference tree(hereafter “ctree”) model to predict the “classe” variable in the dataset, with 10-fold cross validation. It could be simply implemented by the following commands using 3rd party “caret” library. It took about 7 minutes to train the model with my laptop, which had an Intel i5-6200U CPU.

```
trControl<-trainControl(method='cv',number=10)
set.seed<-5566
ctree<-train(classe~.,method='ctree',data=data2, trControl=trControl)
```

Result

The performance of the ctree modle on the training dataset could be assessed with the confusionMatirx command:

```
result<-confusionMatrix(classe,predict(ctree,data2))
overall<-as.data.frame(round(result$overall,digits=4))
names(overall)=c('value')
result$table
```

```
##           Reference
## Prediction    A    B    C    D    E
##          A 5314   94   16   36   11
##          B  118 3431   64   61   44
##          C   32  129 3137   34   20
##          D   47   69   78 2916   37
##          E   18   51   35   44 3380
```

```
overall
```

```
##                 value
## Accuracy       0.9460
## Kappa          0.9316
## AccuracyLower  0.9427
## AccuracyUpper  0.9491
## AccuracyNull   0.2877
## AccuracyPValue 0.0000
## McnemarPValue  0.0000
```

The accuracy of the ctree model was about 94% in this study, which might be considered as acceptible. However, it was a optimistic predication of accuracy, since it did not take out-of-sample error into assessment. A more practicle assessment of accuracy could be obtained from the model object itself.

```
ctree$result
```

```
##   mincriterion  Accuracy     Kappa  AccuracySD    KappaSD
## 1         0.01 0.8983157 0.8713295 0.008382314 0.01061695
## 2         0.50 0.8983678 0.8713896 0.008313016 0.01052800
## 3         0.99 0.8933714 0.8649930 0.008814244 0.01118729
```

As showed, a practicle assessment of accuracy was about 89%.

Prediction

The prediction of behavior in the test dataset could be obtained with the following codes.

```
testData<-read.csv('data/testing.csv',na.strings =c('NA',"",'#DIV/0!'))
testData2<-clean(testData)
pred<-predict(ctree,testData2)
pred
```

```
##  [1] B A B A A E D E A A A C B A E E A B B B
## Levels: A B C D E
```